(2) B.S.

[EA 20737]

TECHNICAL REPORT RG-76-31

BANG-BANG SEEKER MODEL AND MODEL REDUCTION FOR DIGITAL SIMULATION

R. E. Yates and R. P. Wasky
Guidance and Control Directorate
US Army Missile Research, Development
and Engineering Laboratory
US Army Missile Command
Redstone Arsenal, Alabama 35809

5 November 1975

U.S. ARMY MISSILE COMMAND
Redstone Arsenal, Alabama

FORM AMSMI-1021, 1 DEC 65 PREVIOUS EDITION IS OBSOLETE

DISPOSITION INSTRUCTIONS

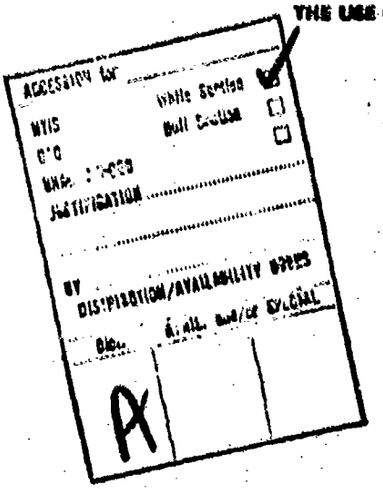
DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL INDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (14) Technical Report RG-76-31	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) BANG-BANG SEEKER MODEL AND MODEL REDUCTION FOR DIGITAL SIMULATION		5. TYPE OF REPORT & PERIOD COVERED (9) Technical rept.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) R. E. Yates R. P. Wasky		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Commander, US Army Missile Command Attn: AMSMI-RG Redstone Arsenal, Alabama 35809		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (DA) 1M263310D074 AMC Management Structure Code No. 633310.12.20400 (12)
11. CONTROLLING OFFICE NAME AND ADDRESS (11) Commander, US Army Missile Command Attn: AMSMI-RPR Redstone Arsenal, Alabama 35809		12. REPORT DATE 5 Nov [redacted] 75 (14p.) 13. NUMBER OF PAGES 39
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)* (16) DA-1-M-263310-D-074		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Terminal Homing Accuracy Demonstration
Runge-Kutta
Polynomial Simplification
Digital Simulation
Transfer Function Model Reduction

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

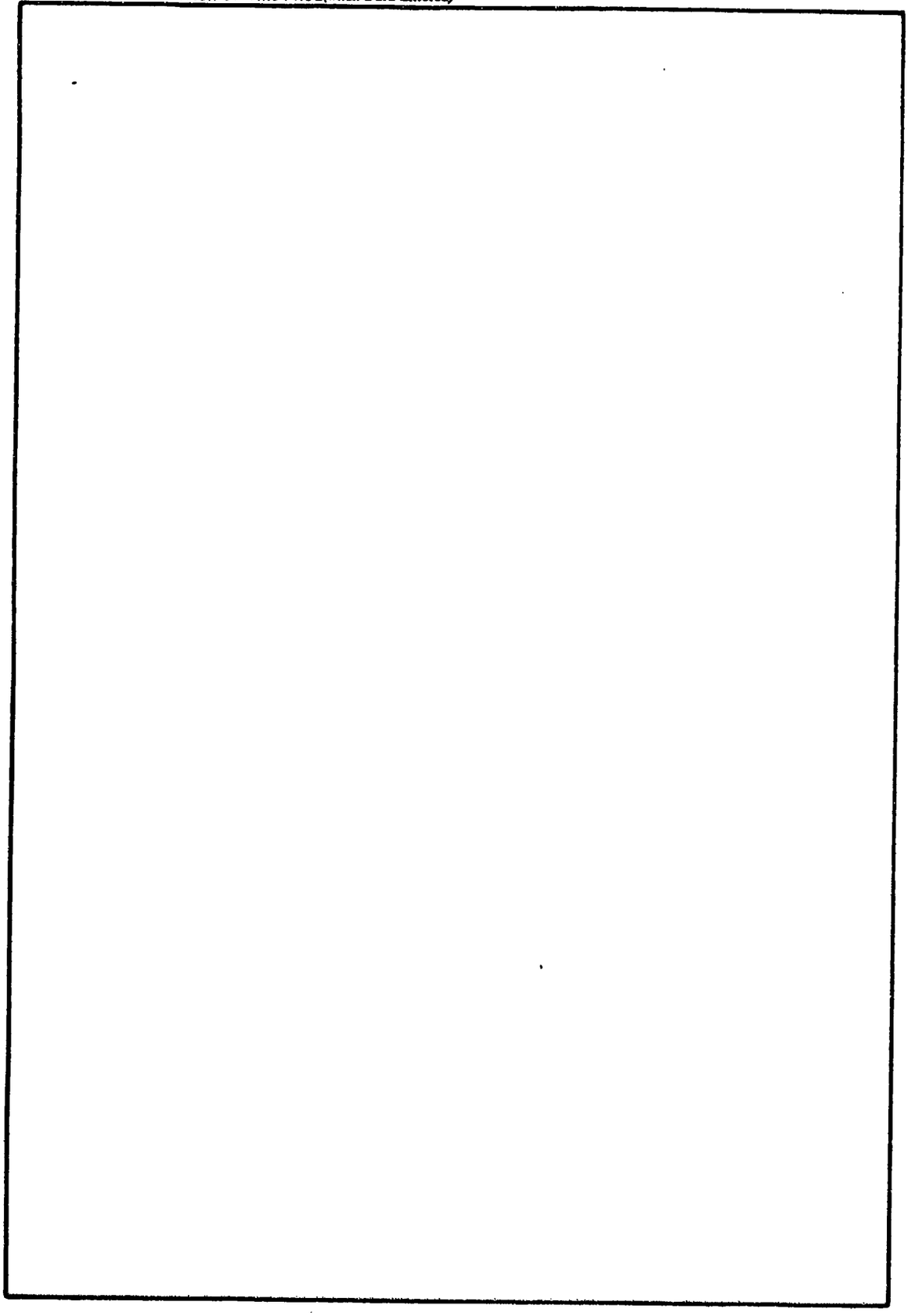
Four computer programs are presented for the analysis of a sampled data terminal homing missile seeker. A Runge-Kutta program is given for the solution of a general dynamical system. Two convenience programs for obtaining the products of a system of polynomials and a line printer plot routine are included. A program, using methods developed by Chen and Shieh, for obtaining a reduced order model from a higher order system is also presented. The programs are illustrated via the study of a Wide Field-of-View (WFOV) laser guided semi-active terminal seeker.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

400 469



CONTENTS

1. INTRODUCTION

A computer simulation of a two rate gyro stabilized platform, laser semi-active seeker is given here. The seeker, shown in Figures 1 and 2, is the Wide Field-of-View (WFOV) seeker developed for use in the Terminal Homing Accuracy Demonstration (THAD) program. A thirteenth order seeker dynamic model is studied and compared to actual seeker performance. The method of Chen and Shieh[1] is used to simplify this model to a second order system. The simplified model was then compared to: (1) the thirteenth order model, (2) a contractor third order model, and (3) the actual system response. The agreement between the responses was quite good and is discussed.

This report emphasizes the digital programs used in this study since they can be used for a wide variety of engineering applications. These routines include a Runge-Kutta numerical analysis method, a plotter package, a compound polynomial simplification process and a transfer function order reduction program. A brief description of each program follows with a sample problem illustration where appropriate.

2. RUNGE-KUTTA DIFFERENTIAL EQUATION SOLUTION

To obtain the seeker response in the track mode, a fourth order Runge-Kutta algorithm for the thirteenth-order servo system was used. Figure 3 shows the differential equation solution program for the seeker. All initial conditions of the state variables XN(1) through XN(13) were set to zero radians. The program was set up to print values of the state variable in degrees every 10 msec using a Runge-Kutta time increment of $h = 10^{-4}$ second.

To measure the seeker's tracking rate, a laser radiation source is set in a position which is 3° from the seeker's caged line-of-sight. In the simulation this angular error (ALOS = 3°) is initially read into the program. Then at time t = 0 the seeker is switched into the track mode permitting a track command signal (SAMP = $\pm$ 2v) to align the sensor head with the source. At t = 45 ms, SAMP = 0 for a period of 5 ms in a manner identical to the seeker's signal processor. Then at t = 50 ms the seeker compares the ALOS error to the platform position XH1(1) to determine whether SAMP will be positive or negative. This comparison continues to occur at 50 ms intervals to update the seeker head position.

[1] C. F. Chen and L. S. Shieh, "A Novel Approach to Linear Model Simplification," Int. J. Control, Vol. 8, 1968, pp. 561-570.

The state equations FX(1) through FX(13) are taken from Figure 2, and initially set equal to zero. The state equations are defined as:

$$X(1) = x$$

$$FX(1) = dx/dt$$

$$\vdots \qquad \vdots$$

$$FX(12) = d^{12}x/dt^{12}$$

$$FX(13) = d^{13}x/dt^{13}$$

Drift rates are entered simply by adding a term to the twelfth state equation such that

FX(12) = 1.E5/3.77 * (-8.58E-3 * X(12) X(11) + X(2) + DRIFT)

Similarly by making changes in the state equations in accordance with the constants of Figure 1, either the pitch or yaw response can be observed. The pitch response will be discussed throughout this report although both cases were studied and found to be essentially identical.

Data cards are read into the programs as follows:

1. IPPS - Pulses per second rate of laser illuminator (IPPS = 20).

2. N - Order of differential equation (N = 13).

3. ALOS - Angular line-of-sight error in degrees (ALOS = 3.0).

4. TN, H, XN(1) ... XN(13) - Initial time (TN = 0), Runge-Kutta increment (H = 10^{-4}), Initia' state variable conditions in radians (XN(1) ... XN(13) are all 0).

These values are printed by the program for reference, as well as the calculated sampling period (TSAM = .050 second) for SAMP sign changes.

3. PLOT PACKAGE

An easily used plot package is included in the Runge-Kutta analysis program to provide a graphic recording of the system response (Figure 4). This routine provides a neatly labeled plot of up to five functions of 501 points each versus time, complete with grid lines and point-by-point print out of the first two functions alongside the graph.

To use the plotter simply call PLOT (Y, M, NF, NS).

Y = the array of data to be plotted. The first function's data is in the form Y(1,1), Y(2,1) ... Y(NF,1), with the second function (if any is to be plotted on the same graph) similarly arranged as Y(1,2), Y(2,2) ... Y(NF,2), and so on with as many as five functions.

M = number of functions per graph (up to 5).

NF = number of points per function beginning with the initial conditions (up to 501 points).

NS = maximum upper limit of the ordinate axis.

It is important to remember that this routine plots from NS - 100 to NS (e.g., if NS = 50 the ordinate is labeled from -50 to +50; NS = 100 the axis is from 0 to 100).

The initial conditions of the platform output XN1(1) and SAMP are loaded into the data array elements Y(1,1) and Y(1,2) early in the Runge-Kutta program at lines 53 and 54, while the remaining values of these two functions are called into the plot program by C(KK+1,1) and C(KK+1,2) at lines 126 and 127.

The plot parameters used in the Runge-Kutta program are given as,

M = 2 for the two functions to be plotted versus time on the same graph, XN1(1) and SAMP respectively.

NF = 401 for a total of 401 points per function.

NS = 50 normally labels the graph from -50 to +50. A discussion of the use of NS with scaled functions will be given in the next section.

Some computer systems, such as the CDC 6600, may not properly execute the plot program unless the Y data array dimension statements are the same in both the main program and the plot subroutine. Furthermore, the Y array may have to be dimensioned Y(5,501) rather than Y(501,5) depending again on the system being used. Of course any desired increase in the maximum number of functions or points to be plotted can be made by simply increasing the data array dimension statements in the two programs.

4. USE OF THE PLOT ROUTINE WITH SCALE FACTORS

Any function(s) may be scaled for clearer graphical display by the plot subroutine. This requires the simple change of only the plot data array cards and an ordinate axis labeling card within the subroutine. A description of the steps used to scale the platform position XN1(1) follows which serves as a guide for any scaling that may be desired.

(1) Referring to lines 53 and 126 of the Runge-Kutta program, the platform angular position was multiplied by a scale factor of 10 to show its relationship to SAMP more adequately. Thus the data array cards for the initial condition and all subsequent values of XN1(1) become respectively

$$C(1,1) = 10. * XN1(1)$$

and

$$C(KK + 1,1) = 10. * XN1(1)$$

(2) Line 18 in the plot routine was changed to label the ordinate axis according to the scaled platform function. Normally this line would read

$$101\ L(I) = 10 * I - 110 + NS$$

labeling the ordinate axis from NS - 100 to NS. However, since XN1(1) was increased by a factor of 10, the axis scale had to be reduced by the same factor. Thus line 18 becomes

$$101\ L(I) = (10 * I - 110 + NS)/10.$$

which now labels the ordinate from (NS - 100)/10 to NS/10. Note that NS = 50 in the main program since the scaled values of XN1(1) range from 0° to approximately 30°. The change in line 18 of the plotter now expands the ordinate from -5 to +5.

(3) Line 52 in the plot routine rescales XN1(1) to its original values for printing along side the graph. Thus,

$$Y(N,L) = Y(N,1)/10.$$

Note that SAMP was not scaled to a larger value and therefore does not correspond to the ordinate label. However this is unimportant since only the sign of SAMP rather than its magnitude is of interest here.

5. POLYNOMIAL SIMPLIFICATION ROUTINE

It is often a long and tedious process to obtain the total polynomial transfer function of complex systems. Many times the simplification of such systems is left in a form which is a long combination of polynomial sums and products. The following program (Figure 5) is a useful means of executing the addition and multiplication of these polynomials in order to produce a transfer function of the form

$$F(s) = \frac{P_m s^m + P_{m-1} s^{m-1} + \ldots + P_1 s + P_0}{Q_n s^n + Q_{n-1} s^{n-1} + \ldots + Q_1 s + Q_0}$$

Let the system transfer function be in the general form of a quotient of two compound polynomials

$$F(s) = \frac{(A_1 + A_2 + \ldots + A_M)(B)}{(C_1 + C_2 + \ldots + C_N)(D)}$$

where A_1, A_2 ... A_M, B, C_1, C_2 ... C_N, D are themselves products of polynomials and constant factors. For example, A_1 might be a product of two polynomials and two constant factors such that

$$A_1 = K_1 K_2 (a_{13} s^3 + a_{12} s^2 + a_{11} s + a_{10})(a_{21} s + a_{20})$$

The operation of this program is rather straightforward. First the A polynomials are solved (if any are present), then added together and finally their sum is multiplied times the B polynomial product. The resultant numerator is printed giving the order of "s" and its coefficient. The denominator polynomials are handled in the same manner.

The following sample problem illustrates the use of data cards for this program.

Let

$$F(s) = \frac{5(s^4 - 2s^3 + 2s^2 + s - 1)(s^2 + 2)}{[(s^2 + 2s - 3)(s + 2) + 20(s^3 - 2)(s + 1)](s^2 - 3s + 1)}$$

where

$$B = 5(s^4 - 2s^3 + 2s^2 - 1)(s^2 + 2)$$

$$C_1 = (s^2 + 2s - 3)(s + 2)$$

$$C_2 = 20(s^3 - 2)(s + 1)$$

$$D = s^2 - 3s + 1$$

Data cards are read in the following order beginning with the A polynomials if any are present (Figure 6).

1.	NA	Number of A polynomials which must be added. Since there are no A polynomials here, NA = 0 and we now consider the B polynomial.
2.	NP, KK	Number of polynomials to be multiplied in B and the number of constant factors respectively. Here, NP = 2 and KK = 1.
3.	K(I)	Constant factors in B (other than 1.0). If no constants are present, simply omit this data card.
4.	NI	Order of the first polynomial to be multiplied in B. Since the first polynomial is fourth order, NI = 4.
5.	POLY 1(I)	Coefficients of the first polynomial factor beginning with the highest order "s" coefficient. This card reads: 1.E0 -2.E0 2.E0 1.E0 -1.E0).
6.	N2	Order of second polynomial factor in B. N2 = 2.
7.	POLY 2(I)	Coefficients of the second polynomial. This card is 1.E0 0.E0 2.E0.
8.	NA	Number of C polynomials in the denominator. NA = 2.
9.	NP, KK	Refers to C_1. Here NP = 2 and KK = 0.
	K(I)	Since KK = 0, this card is omitted.
10.	NI	Order of first polynomial to be multiplied in C_1. NI = 2.
11.	POLY 1(I)	Refers to C_1.

12.	NZ	Refers to C_1.
13.	POLY 2(I)	Refers to C_1.
14.	NP, KK	Refers to C_2.
15-19.		Refers to C_2.
20.	NP, KK	Here NP = 2 (includes the C polynomial sum and the D factors). KK = 0.
	K(I)	Since KK = 0, this card is omitted.
21.	N2	Order of the first D polynomial factor. POLY 1(I) is the sum of the C polynomials which is stored in the program. No cards are needed for POLY 1(I) following the addition of A or C polynomials. Here the first D polynomial is second order, so N2 = 2.
22.	POLY 2(I)	Coefficients of the first D polynomial factors. This card is 1.E0 -3.E0 1.E0.

Referring to the seeker block diagram it is apparent that the transfer function up to the platform rate output is of the twelfth order. In some cases it may be of interest to monitor the tracking rate of the platform as well as its actual position. For this reason a twelfth order system polynomial was obtained using this program and the data cards in Figure 7. Finally, it became desirable to reduce this rate portion of the system to a second order approximation model for easier design manipulation. The following program was used to this effect.

6. REDUCTION IN ORDER FOR A DIFFERENTIAL EQUATION

This program reduces the order of an n^{th} order differential equation by expanding its s-plane polynomial into a continued fraction and truncating certain of its quotients[1]. The remaining expansion quotients are then used to write all the lower order polynomials beginning with order n - 1.

The high order polynomial is initially arranged in ascending order, beginning with the constant terms.

$$F(s) = \frac{P_0 + P_1 s + P_2 s^2 + \ldots + P_m s^m}{Q_0 + Q_1 s + Q_2 s^2 + \ldots + Q_n s^n}$$

From this polynomial the program calculates and prints the 2n continued fraction terms (e.g., a 12^{th} order equation has 24 terms).

$$F(s) = \cfrac{1}{H_1 + \cfrac{1}{\frac{H_2}{s} + \cfrac{1}{H_3 + \cfrac{1}{\frac{H_4}{s} + \cfrac{1}{\ddots + \cfrac{1}{\frac{H_{24}}{s}}}}}}}$$

To derive an approximation of order ω, only the first 2ω terms are retained for a reversal of the expansion process which yields the lower order polynomial. For a second order model of the twelfth order seeker platform system, the first four continued fraction terms were saved so that

$$F(s) = \cfrac{1}{34.474 + \cfrac{1}{\frac{-6.8275}{s} + \cfrac{1}{-3.057 + \cfrac{1}{\frac{20.732}{s}}}}}$$

which produced

$$F(s) = \frac{13.904s + 432.71}{s^2 + 415.96s + 14917.6}$$

The program is designed to print the terms of the original system and all of its lower order approximations in ascending order.

Data is easily read into this program as Figure 9 demonstrates. The only restriction is that the denominator is assumed to be one order higher than the numerator for execution of the continued fraction process. This condition is satisfied by the introduction of "dummy" data terms. For instance the WFOV seeker platform polynomial has a 9th order numerator and a 12th order denominator such that

$$F(s) = \frac{P_0 + P_1 s + \ldots + P_9 s^9}{Q_0 + Q_1 s + \ldots + Q_{12} s^{12}}$$

To make the numerator one order less than the denominator simply enter zero for the numerator s^{10} and s^{11} coefficients so that

$$F(s) = \frac{P_0 + \quad + \ldots + P_9 s^9 + 0s^{10} + 0s^{11}}{Q_0 + Q_1 s \quad \ldots + Q_{10} s^9 + Q_{10} s^{10} + Q_{11} s^{11} + Q_{12} s^{12}}$$

Referring to Figure 9, data is entered for the seeker platform as follows.

1. NN - number of terms in the numerator. NN = 12.

2. A(1) ... A(NN) - coefficients of the numerator in ascending order beginning with the constant term. Note that A(11) and A(12) are dummy (zero) terms.

3. B(1) ... B(ND) - coefficients of the denominator in ascending order beginning with the constant term ND = NN + 1 (reference line 19 of the Figure 8 program listing).

7. RESULTS

The thirteenth order model of the WFOV seeker yielded an accurate simulation of the seeker's time response to a 3° line-of-sight error. As expected, the model demonstrated a 3°/sec tracking rate with an input forcing function (SAMP) correction every 50 ms (Figure 10). The state equation set up for the seeker is given in Figure 2 as equations FX(1) through FX(13), where $FX(1) = \frac{dx}{dt}$, $FX(2) = \frac{d^2x}{dt^2}$, etc.

A polynomial transfer function was then derived from the servo block diagram and is given as

$$F(s) = \frac{p_9 s^9 + p_8 s^8 + p_7 s^7 + p_6 s^6 + p_5 s^5 + p_4 s^4 + p_3 s^3 + p_2 s^2 + p_1 s^1 + p_0}{q_{12} s^{12} + q_{11} s^{11} + q_{10} s^{10} + q_9 s^9 + q_8 s^8 + q_7 s^7 + q_6 s^6 + q_5 s^5 + q_4 s^4 + q_3 s^3 + q_2 s^2 + q_1 s^1 + q_0} \cdot \frac{1}{s}$$

where

$q_{12} = 1.265(10^{-23})$

$q_{11} = 3.327(10^{-19})$

$q_{10} = 2.084(10^{-15})$

$p_9 = 1.556(10^{-15})$ $\quad q_9 = 5.703(10^{-12})$

$p_8 = 3.340(10^{-11})$ $\quad q_8 = 7.968(10^{-9})$

$p_7 = 8.450(10^{-8})$ $\quad q_7 = 5.966(10^{-6})$

$p_6 = 7.229(10^{-5})$ $\quad q_6 = 2.440(10^{-3})$

$p_5 = 2.424(10^{-2})$ $\quad q_5 = 7.930(10^{-1})$

$p_4 = 4.811$ $\quad q_4 = 1.625(10^{2})$

$p_3 = 5.802(10^{2})$ $\quad q_3 = 1.965(10^{4})$

$p_2 = 4.495(10^{4})$ $\quad q_2 = 1.493(10^{6})$

$p_1 = 1.909(10^{6})$ $\quad q_1 = 6.116(10^{7})$

$p_0 = 3.178(10^{7})$ $\quad q_0 = 1.095(10^{9})$

The twelfth order polynomial represents the system up to the platform rate output while the final $\frac{1}{s}$ integrator provides the platform position (reference Figures 1 and 2).

The twelfth order polynomial was reduced to the second order approximation

$$F(s) = \frac{13.904s + 432.71}{s^2 + 415.96s + 14917.6}$$

and is shown in Figure 11 as part of a third order system model. The state equations for Runge-Kutta integration of this system are

$$FX(3) = SAMP - 415.96\ X(3) - 14917.6\ X(2)$$

$$FX(2) = X(3)$$

$$FX(1) = 13.904\ X(3) + 432.71\ X(2)$$

The third order approximation accurately demonstrated a 3°/sec tracking rate, however its forced platform sweep generally remained below a 3° position and was equal in magnitude but 180° out of phase from the higher order system (Figure 10).

The discrepancy between the two models was due to the slight lead of the third order system. For instance at time t = 1.1 second the platform positions of the thirteenth and third order systems were 2.99992 and 3.00528 respectively. Consequently SAMP remained positive for the first system but changed to a negative value for the latter system causing it to decrease in value. This out of phase relationship maintained itself throughout the entire forced switching mode.

Dispite the above-mentioned minor differences the lower order model as a whole proved to be a sufficiently accurate representation of the thirteenth order system. This confirms the validity of the continued fraction expansion process used in the fourth computer program to derive lower order systems.

A third order model provided by the seeker vendor (Figure 12) had a slower 2.71°/sec tracking rate. However after the seeker platform reached the 3° position at 1.1 seconds, its performance was very close to that of the thirteenth order system (Figure 10).

The state equations for this approximation are

$$FX(3) = .4(11700(SAMP - X(2)) - X(3))$$

$$FX(2) = X(3) + .026316\ FX(3)$$

$$FX(1) = X(2)$$

It should be noted that the state equations and state variables for the vendor system are already given in units of degrees rather than in radian measure which was used for the other two models.

The introduction of both constant and time-varying drift rates up to 0.1°/sec peak value had little effect on the models' responses. Greater drift rates, however, caused failure of the SAMP forcing function for the 13^{th} order model to switch signs at the end of every 50 ms interval (multiple pulsing) as well as a reduction in the tracking speeds of that system. The vendor model was not adversely affected

even with drift rates as large as 5°/sec. In actual system performance the seeker would be unable to cope with such large drifts, which indicates the failure of the vendor model to properly simulate internal friction effects.

Drift was entered into the 13^{th} order system by the state equation

$$FX(12) = \frac{10^5}{3.77} (-.00858\ X(12) - X(11) + X(2) + DRIFT)$$

and for the vendor model

$$FX(2) = X(3) + .026316\ FX(3) + DRIFT$$

The in-house third order approximation was not considered for drifting since its response had so closely approached that of the thirteenth order system.

8. CONCLUSIONS

A description of four computer programs for the simplification and analysis of a discrete data tracking system has been presented. These programs are demonstrated via the solution of the step response of a laser semi-active seeker. The programs are general and should require minor modification for study of other seeker systems.

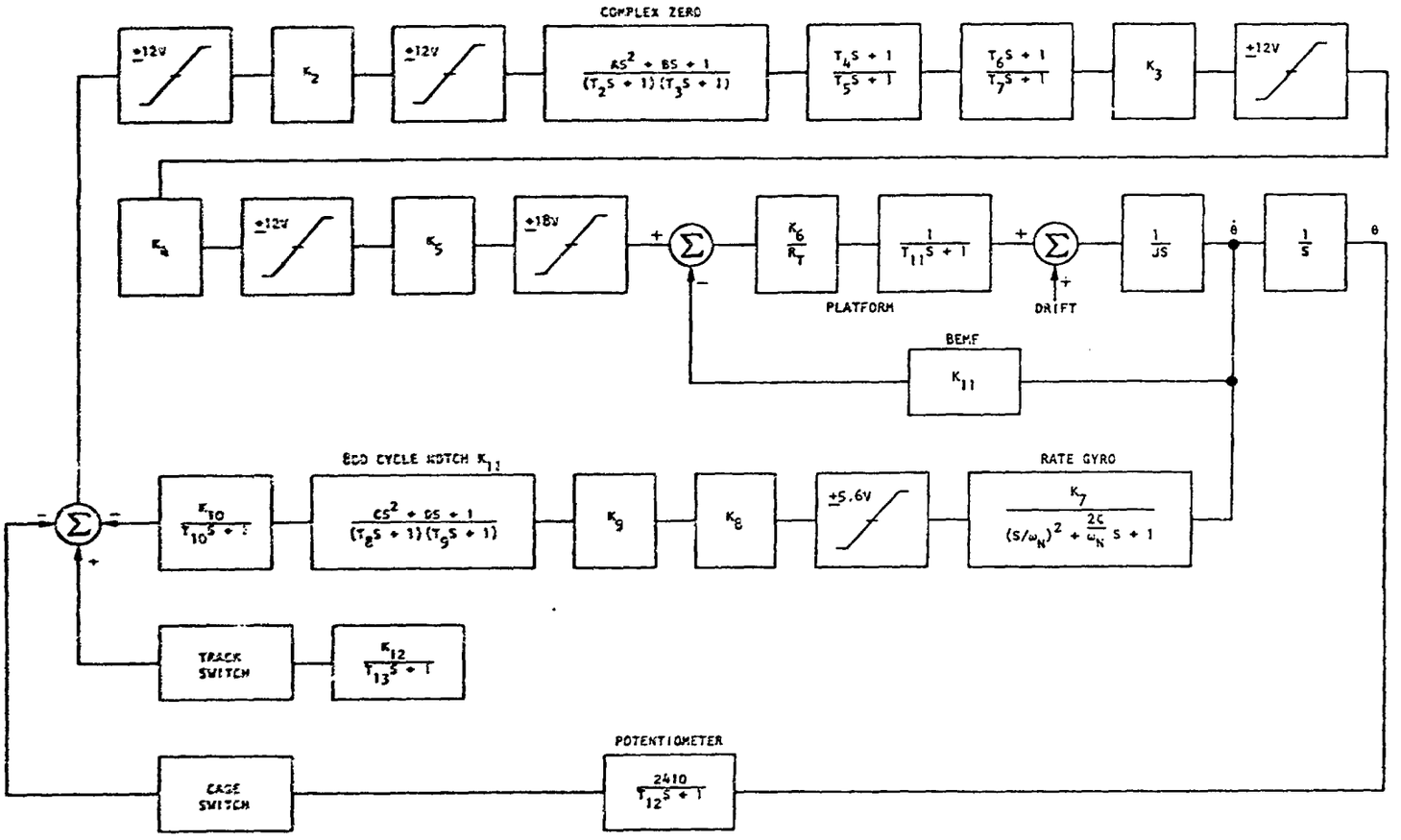


FIGURE 1. DETAILED SERVO BLOCK DIAGRAM

Constants

$A = 7.7 \times 10^{-5}$ sec^2

$B = 8.52 \times 10^{-3}$ sec

$C = 4.03 \times 10^{-8}$ sec^2

$D = 1.205 \times 10^{-6}$ sec

$T_2 = 4.93 \times 10^{-3}$ sec

$T_3 = 4.97 \times 10^{-4}$ sec

$T_4 = 1.785 \times 10^{-2}$ sec

$T_5 = 5.92 \times 10^{-3}$ sec

$T_6 = 2.33 \times 10^{-2}$ sec

$T_7 = 2.5$ sec

$T_8 = 7.22 \times 10^{-4}$ sec

$T_9 = 5.35 \times 10^{-5}$ sec

$T_{10} = 1.05 \times 10^{-3}$ sec

$T_{11} = 4 \times 10^{-4}$ sec

$1/\omega_n^2 = 3.77 \times 10^{-5}$ sec^2

$2\zeta/\omega_n = 8.58 \times 10^{-3}$ sec

$\omega_n = 163$ 1/sec*

$T_{12} = 0.033$ sec

$K_1 = 0.405$ v/v

$K_2 = 1$ v/v

$K_3 = 1.52$ v/v

$K_4 = 9.4$ v/v in pitch, 20 v/v in yaw

$K_5 = 10$ v/v

$K_6 = 5.56 \times 10^5$ (dyne cm)/amp

$K_7 = 7.98$ vrms/rad/sec

$K_8 = 1.2$ v/v

$K_9 = 0.9$ v/vrms

$K_{10} = 1.6$ v/v

$K_{11} = 0.055$ v/rad/sec

K_{ALL} = product of all rate loop gains = (1.25×10^8) for pitch, 2.51×10^8 for yaw)

$R_T = 8.8$ ohms

$J = 2.27 \times 10^4$ gm cm in yaw, 1.08×10^4 gm cm in pitch

$K_{12} = .4$ v/v

$T_{13} = 6 \times 10^{-3}$ sec

FIGURE 1a. CONSTANTS FOR FIGURE 1

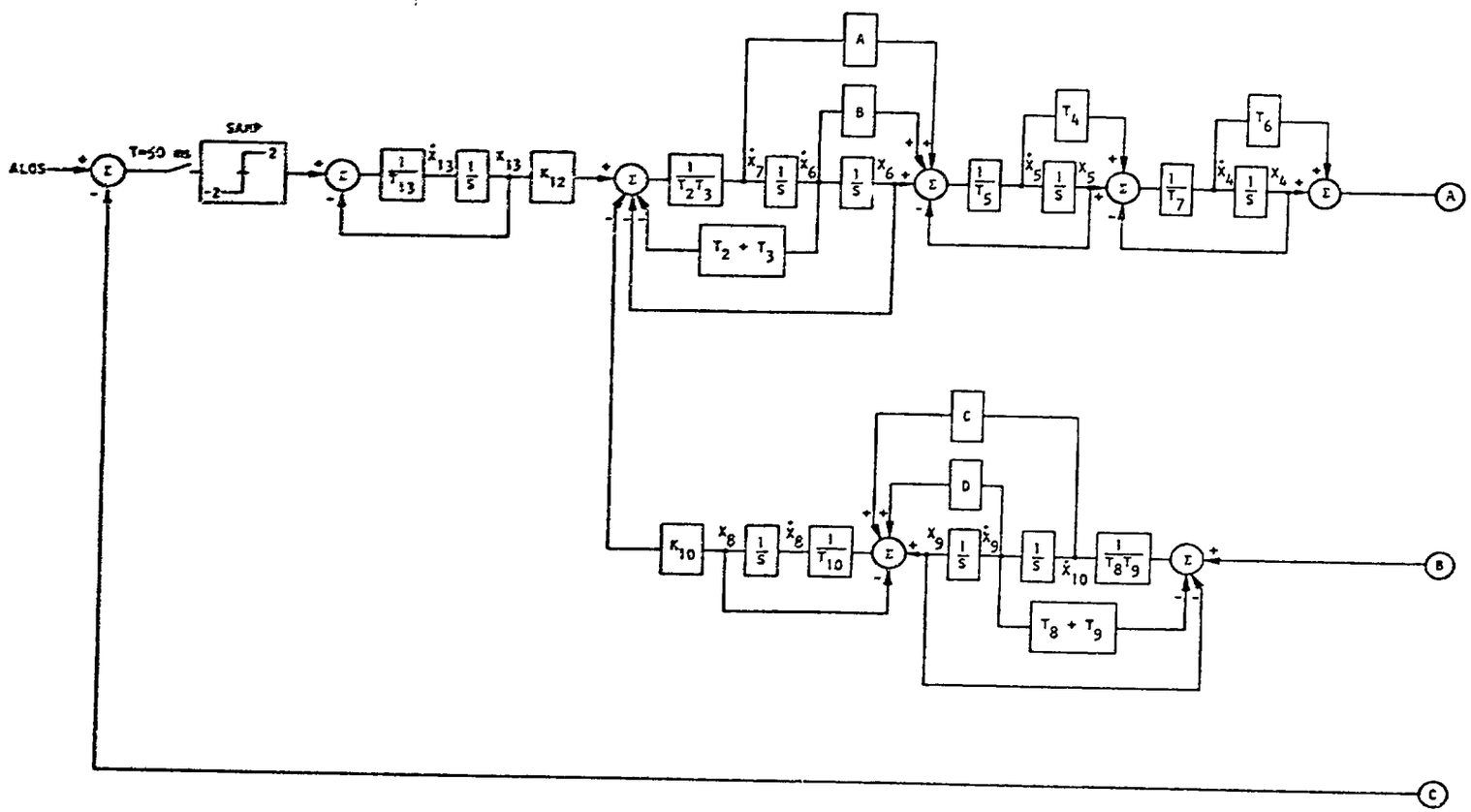


FIGURE 2. STATE VARIABLE SET-UP OF THE LOOP SERVO SYSTEM

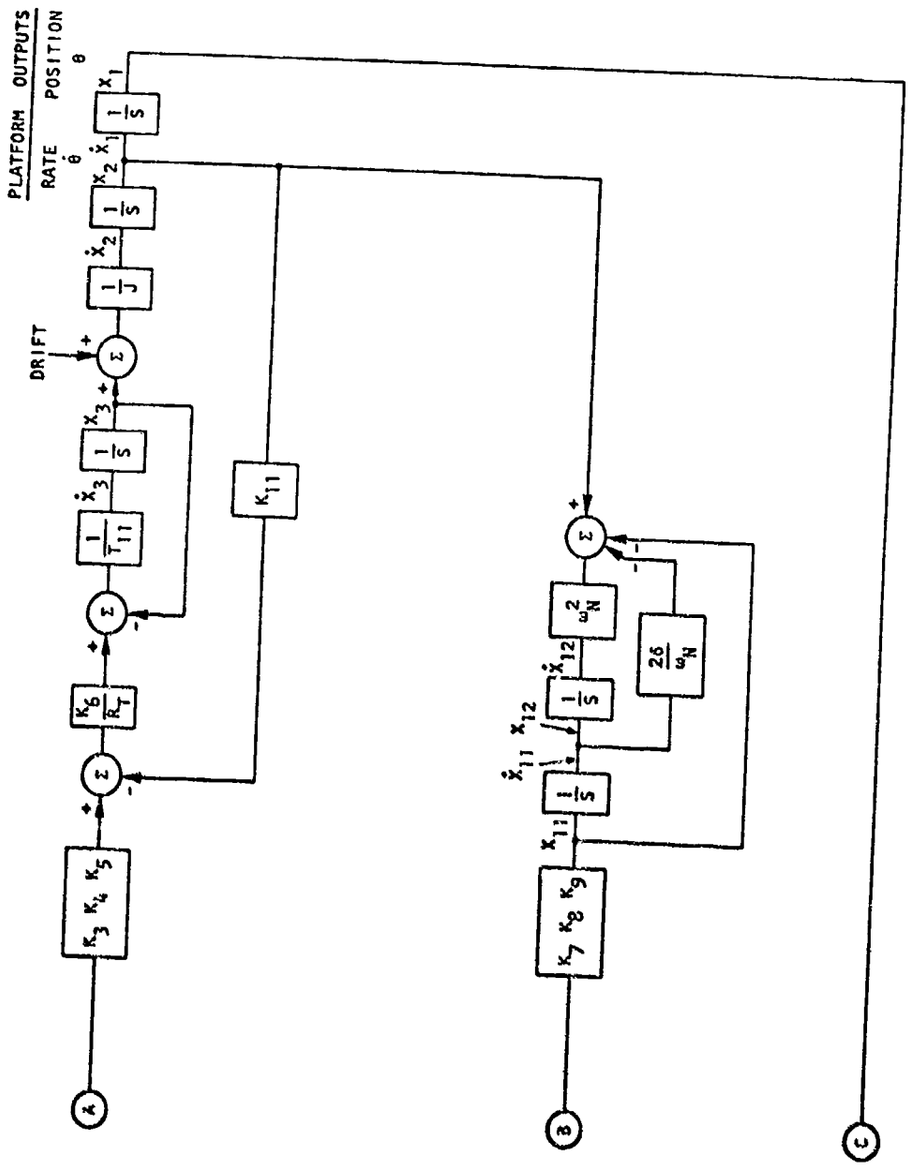


FIGURE 2. STATE VARIABLE SET-UP OF THE LOOP SERVO SYSTEM (Continued)

```
C     SOLUTION OF A DIFFERENTIAL EQUATION USING THE RUNGE-KUTTA METHOD
C
C     IPPS = PULSE REPETITION RATE (PPS)
C     TSAM = SAMPLING PERIOD (SECONDS)
C     N = ORDER OF DIFFERENTIAL EQUATION
C     ALOS = ANGULAR LINE OF SIGHT ERROR (DEGREES)
C     TN = INITIAL TIME (SEC)
C     H = RUNGE-KUTTA TIME INCREMENT (SEC)
C     XN(1)...XN(N) = STATE VARIABLE INITIAL CONDITIONS (RADIANS)
C     FX(1)...FX(N) = STATE EQUATIONS (RADIANS)
C
      DIMENSION XN(20),X(20),Q(20,20),FX(20),ICOUNT(20),XN1(20),Y(501,5)
   68 FORMAT(22X,6(3HXN(,I2,1H),9X),4HSAMP)
   69 FORMAT(16X,1P7E15.6)
   70 FORMAT(1H1)
   81 FORMAT(2I3)
   83 FORMAT(5E15.7)
   86 FORMAT(1H,1PE14.5,1P7E15.6)
   87 FORMAT(///7X,4HTIME,10X,7(3HXN(,11,1H),10X))
      READ(5,81)IPPS
      READ(5,81)N
      READ(5,83) ALOS
      READ(5,83) TN,H,(XN(NN),NN=1,N)
      WRITE(7,70)
      WRITE(7,81)IPPS
      WRITE(7,81) N
      WRITE(7,86) ALOS
C     CONVERT XN (RADIANS) TO XN1 (DEGREES)
      DO 55 IQ = 1,N
   55 XN1(IQ) = XN(IQ) *57.2958
      WRITE(7,86)TN,H,(XN1(NN),NN=1,N)
      NCOUN=0
      PPS=FLOAT(IPPS)
      TSAM=1./(PPS*H)
      WRITE(7,86) TSAM
C     DEFINE THE INPUT FORCING FUNCTION
      IF( ALOS - 57.2958*XN(1)) 802,802,803
  802 SAMP = -2.0
      GO TO 804
  803 SAMP = 2.0
C     WRITE THE HEADING AND INITIAL CONDITIONS
      DO 88 IX=1,N
   88 ICOUNT(IX)=IX
      WRITE(7,87)(ICOUNT(IX),IX=1,7)
      WRITE(7,68)(ICOUNT(IX),IX=8,13)
      WRITE(7,86)TN,(XN1(NN),NN=1,7)
      WRITE(7,69)(XN1(N1),N1=8,13),SAMP
C     CLEAR THE STATE EQUATIONS
  804 DO 5 M=1,N
    5 FX(M)=0.0
C     SOLVE THE DIFFERENTIAL EQUATION
C     THIS LOADS THE INITIAL CONDITIONS INTO THE Y DATA ARRAY
      Y(1,1)=10.*XN1(1)
      Y(1,2)=SAMP
      Y(1,3)=X(3)+.5*SAMP
```

FIGURE 3. LISTING OF THE RUNGE-KUTTA SOLUTION OF THE THIRTEENTH ORDER MISSILE SEEKER SYSTEM

```
C       RUN PROGRAM UNTIL TIME TN = H*KK*KW  SECONDS
        DO 201 KK=1,400
C       PROGRAM PRINTS EVERY TIME TN = H*KW  SECONDS
        DO 200 KW=1,100
      1 L=1
        T=TN
C       SET STATE VARIALBES EQUAL TO THEIR ITERATION VALUES
        DO 777 K=1,N
    777 X(K)=XN(K)
        GO TO 101
     10 DO 151 K=1,N
    151 Q(K,L)=H*FX(K)
        T=TN+H/2.
        DO 252 K=1,N
    252 X(K)=XN(K)+Q(K,L)/2.
        L=2
        GO TO 101
     20 DO 251 K=1,N
    251 Q(K,L)=H*FX(K)
        T=TN+H/2.
        DO 352 K=1,N
    352 X(K)=XN(K)+Q(K,L)/2.
        L=3
        GO TO 101
     30 DO 351 K=1,N
    351 Q(K,L)=H*FX(K)
        T=TN+H
        DO 452 K=1,N
    452 X(K)=XN(K)+Q(K,L)
        L=4
        GO TO 101
     40 DO 451 K=1,N
    451 Q(K,L)=H*FX(K)
        GO TO 7
C       STATE EQUATIONS FOLLOW
    101 FX(13)=(-X(13)+SAMP)/.006
        FX(12)=1.E5/3.77*(-8.58E-3*X(12)-X(11)+X(2))
        FX(11)=X(12)
        FX(10)=1.E8/3.8627*(-7.755E-4*X(10)-X(9)+.9*1.2*7.98*X(11))
        FX(9)=X(10)
        FX(8)=1.E3/1.05*(-X(8)+4.03E-8*FX(10)+1.205E-6*FX(9)+X(9))
        FX(7)=1.E7/24.502*(-5.427E-3*X(7)-X(6)-1.6*X(8)+.4*X(13))
        FX(6)=X(7)
        FX(5)=1.E3/5.92*(-X(5)+7.7E-5*FX(7)+8.52E-3*FX(6)+X(6))
        FX(4)=(-X(4)+X(5)+1.785E-2*FX(5))/2.5
        FX(3)=2.5E3*(-X(3)+6.32E4*(-.055*X(2)+1.52*94.0*(X(4)+2.3E-2*FX
       1(4))))
        FX(2)=X(3)/1.08E4
        FX(1)=X(2)
        GO TO (10,20,30,40),L
      7 TN=TN+H
        DO 8 K=1,N
      8 XN(K)=XN(K)+(Q(K,1)+2.*Q(K,2)+2.*Q(K,3)+Q(K,4))/6.
```

FIGURE 3. LISTING OF THE RUNGE-KUTTA SOLUTION OF THE THIRTEENTH ORDER MISSILE SEEKER SYSTEM (Continued)

```
      SAMP=0.
  210 IF (NCOUN .LT. 500) GO TO 200
      IF( ALOS - 57.2958*XN(1)) 902,902,903
  902 SAMP = -2.0
      NCOUN=0
      GO TO 200
  903 SAMP = 2.0
      NCOUN = 0
  200 CONTINUE
C     CONVERT XN (RADIANS) TO XN1 (DEGREES)
      DO 503 IQ = 1,N
  503 XN1(IQ) = XN(IQ) *57.2958
C     PRINT THE FINAL SOLUTION OF THE STATE VARIABLES
      WRITE(7,86)TN,(XN1(NN),NN=1,7)
      WRITE(7,69)(XN1(N11),N11=8,13),SAMP
C     THIS SCALES THE INPUT DATA ARRAY FOR THE PLOT ROUTINE
      Y(KK+1,1)=10.*XN1(1)
      Y(KK+1,2)=SAMP
      Y(KK+1,3)=GUID
  201 CONTINUE
      H=2
      NF=401
      NS=50
      WRITE(7,70)
      CALL PLOT(Y,H,NF,NS)
      STOP
      END
```

FIGURE 3. LISTING OF THE RUNGE-KUTTA SOLUTION OF THE THIRTEENTH ORDER MISSILE SEEKER SYSTEM (Continued)

```
      SUBROUTINE PLOT(Y,M,NF,NS)
C     Y = ARRAY OF DATA TO BE PLOTTED, WHICH IS READ IN THE FORM
C        Y(1,1) ... Y(NF,1), Y(1,2) ... Y(NF,2) ... Y(1,M) ... Y(NF,M)
C     M = NUMBER OF PLOTS
C     NF = NUMBER OF PLOTTED POINTS ALONG ABSCISSA, BEGINNING WITH THE
C        INITIAL CONDITIONS AT TIME T = 0
C     NS = MAXIMUM UPPER LIMIT OF ORDINATE (ROUTINE PLOTS FROM NS-100 TO NS)
C
      DIMENSION Y(501,5),LINE(101),L(11),JL(5)
      DATA JL(1),JL(2),JL(3),JL(4),JL(5)/1HA,1HB,1HC,1HD,1HF/,JN,JP,JI,
     1JBLANK,JZ/1H-,1H+,1H1,1H ,1H$/
      DO 99 I=1,101
   99 LINE(I)=JBLANK
      NN=0
      N=1
C     LABEL THE ORDINATE AXIS
      DO 101 I=1,11
  101 L(I)=(10*I-110+NS)/10.
      IF (M-2) 103,104,104
  103 WRITE(7,105) (L(I),I=1,11)
  105 FORMAT (2X,11(I4,6X),1X,6HY(N,1))
      GO TO 115
  104 WRITE(7,106) (L(I),I=1,11)
  106 FORMAT (2X,11(I4,6X),1X,6HY(N,1),7X,6HY(N,2))
      GO TO 115
  110 IF (NN-10) 125,115,115
  115 ND=0
      NN=0
      DO 120 I=1,10
      ND=ND+1
      LINE(ND)=JP
      DO 120 J=1,9
      ND=ND+1
  120 LINE(ND)=JN
      LINE(101)=JP
      GO TO 135
  125 DO 130 I=1,101,10
  130 LINE(I)=JI
C     CHANGE NUMERICAL DATA TO LETTERS
  135 DO 160 I=1,M
      XNS=NS
      JA=Y(N,1)*10.49999-XNS
      IF (JA-101) 140,155,145
  140 IF (JA) 150,150,155
  145 LINE(101)=JZ
      GO TO 160
  150 LINE(1)=JZ
      GO TO 160
  155 LINE(JA)=JL(I)
  160 CONTINUE
C     THIS RESCALES Y(N,1),Y(N,2) TO ORIGINAL VALUES
      Y(N,1)=Y(N,1)/10.
      NC=(N-1)/10
      IF (M-2) 163,173,173
  163 IF(NN) 165,165,175
```

FIGURE 4. LISTING OF THE PLOTTER PACKAGE

```
      WRITE(7,166) NC,LINE,Y(N,1)
  166 FORMAT(I4,1X,10IA1,1PE13.5)
      GO TO 185
  170 WRITE(7,171) LINE,Y(N,1)
  171 FORMAT(5X,10IA1,1PE13.5)
      GO TO 185
  173 IF (NN) 175,175,180
  175 WRITE(7,176) NC,LINE,Y(N,1),Y(N,2)
  176 FORMAT(I4,1X,10IA1,1P2E13.5)
      GO TO 185
  180 WRITE(7,181) LINE,Y(N,1),Y(N,2)
  181 FORMAT(5X,10IA1,1P2E13.5)
  185 DO 190 I=1,101
  190 LINE(I)=JBLANK
      NN=NN+1
  195 N=N+1
      IF (N-NF) 110,110,200
  200 RETURN
      END
```

FIGURE 4. LISTING OF THE PLOTTER PACKAGE (Continued)

```
C       THIS PROGRAM SOLVES COMPOUND SERVO SYSTEM POLYNOMIALS OF THE FORM
C                 (A1 + A2 + ... + AM)(B)
C                 ----------------------
C                 (C1 + C2 + ... + CN)(D)
C       WHERE A1, A2, ... AM, B, C1. ... CN, D   ARE PRODUCTS OF POLYNOMIALS
C       AND ANY CONSTANT FACTORS.
C       THE PROGRAM FIRST SOLVES THE "A" POLYNOMIALS, ADDS THEM, THEN
C       MULTIPLIES THEIR SUM TIMES THE "B" POLYNOMIAL PRODUCT.
C       THE PROCESS IS REPEATED FOR THE DENOMINATOR
C
C       NA = NUMBER OF "A" OR "C" POLYNOMIAL PRODUCTS TO BE ADDED
C       NP = NUMBER OF POLYNOMIALS TO BE MULTIPLIED
C       KK = NUMBER OF CONSTANT FACTORS
C       N1 = ORDER OF FIRST POLYNOMIAL
C       N2 = ORDER OF SECOND AND SUCCEEDING POLYNOMIALS
C
C       DATA IS READ IN AS CONSTANT COEFFICIENTS K(1),K(2), ETC. WHICH ARE
C       FACTORED OUTSIDE OF THE POLYNOMIALS.
C       THEN AS (N1+1) COEFFICIENTS OF THE FIRST POLYNOMIAL BEGINNING WITH
C       THE HIGHEST ORDER "S" COEFFICIENT.
C       COEFFICIENTS OF SECOND OR SUCCEEDING POLYNOMIALS BEGINNING WITH
C       THE HIGHEST ORDER "S" COEFFICIENT.
C       FINAL POLYNOMIAL IS PRINTED LISTING DEGREE OF S AND ITS COEFFICIENT
C
      DIMENSION A(20,20),POLY1(20),POLY2(20),P(20,20),IT(20)
      REAL KP,K(20)
    1 FORMAT(2I3)
    2 FORMAT(1P8E10.3)
    3 FORMAT(5H  S**,I3,2X,1PE15.6)
    4 FORMAT(19H  POLYNOMIAL NUMBER,I3,8H FOLLOWS)
    5 FORMAT(15H  PRODUCT NUMBER,I3,8H FOLLOWS)
    6 FORMAT(18H  CONSTANTS FOLLOW)
    7 FORMAT(38H  POLYNOMIAL NUMBER  1 TIMES CONSTANTS)
    8 FORMAT(//36H  FINAL POLYNOMIAL NUMERATOR FOLLOWS)
    9 FORMAT(//38H  FINAL POLYNOMIAL DENOMINATOR FOLLOWS)
   10 FORMAT(1H1)
      DO 300 JJJ=1,2
      READ(5,1) NA
      DO 20 I=1,20
      DO 20 J=1,20
   20 P(I,J)=0.
      NB=0
      IF(NA.EQ.0) NA=1
   30 DO 100 JJ=1,NA
      READ(5,1) NP,KK
      IPOLY=1
      KP=1.0
      IF(KK.EQ.0) GO TO 50
      READ(5,2) (K(I),I=1,KK)
      WRITE(6,6)
      DO 40 I=1,KK
      KP=KP*K(I)
   40 WRITE(6,2) K(I)
   50 IF(NB.EQ.1) GO TO 55
```

FIGURE 5. THE POLYNOMIAL SIMPLIFICATION PROGRAM

```
      READ(5,1) N1
      NPA1=N1+1
      READ(5,2) (POLY1(I),I=1,NPA1)
   55 WRITE(6,4) IPOLY
      DO 60 I=1,NPA1
      ID=NPA1-I
   60 WRITE(6,3) ID,POLY1(I)
      WRITE(6,7)
   70 DO 80 I=1,NPA1
      ID=NPA1-I
      POLY1(I)=KP*POLY1(I)
   80 WRITE(6,3) ID,POLY1(I)
      ISUB=NP-1
      DO 100 J=1,ISUB
      READ(5,1) N2
      NPA2=N2+1
      READ(5,2) (POLY2(I),I=1,NPA2)
      IPOLY=IPOLY+1
      WRITE(6,4)IPOLY
      DO 90 I=1,NPA2
      ID=NPA2-I
   90 WRITE(6,3) ID,POLY2(I)
      CALL POLSET (NPA1,NPA2,A,POLY1,POLY2)
      IT(JJ)=NPA1
      IPROD=IPOLY-1
      WRITE(6,5) IPROD
      DO 100 I=1,NPA1
      ID=NPA1-I
      P(ID+1,JJ)=POLY1(I)
  100 WRITE(6,3) ID,POLY1(I)
      IF(NA.EQ.1) GO TO 200
      CALL POLADD (NA,P,IT,POLY1,NPA1)
      NA=1
      NB=1
      GO TO 30
  200 IF(JJJ.EQ.2) GO TO 210
      WRITE(6,8)
      GO TO 220
  210 WRITE(6,9)
  220 DO 230 I=1,NPA1
      ID=NPA1-I
  230 WRITE(6,3) ID,POLY1(I)
      WRITE(6,10)
  300 CONTINUE
      STOP
      END

      SUBROUTINE POLSET (N1,N2,A,POL1,POL2)
      DIMENSION A(20,20),POL1(20),POL2(20)
      NN=N1+N2-1
      DO 11 I=1,N2
      DO 11 J=1,NN
   11 A(J,I)=0.0
      DO 21 I=1,N1
      J=I
      DO 21 K=1,N2
      A(J,K)=POL1(I)
   21 J=J+1
```

FIGURE 5. THE POLYNOMIAL SIMPLIFICATION PROGRAM (Continued)

```
      DO 31 I=1,NN
   31 POL1(I)=0.0
      DO 41 I=1,NN
      DO 41 K=1,N2
   41 POL1(I)=POL1(I)+A(I,K)*POL2(K)
      N1=NN
      RETURN
      END

      SUBROUTINE POLADD (NA,P,IT,ADD,ICOUNT)
      DIMENSION P(20,20),IT(20),ADD(20)
      ICOUNT=IT(1)
      DO 12 I=2,NA
      IF(ICOUNT.GE.IT(I)) GO TO 12
      ICOUNT=IT(I)
   12 CONTINUE
      DO 22 I=1,20
   22 ADD(I)=0.
      DO 32 I=1,ICOUNT
      ID=ICOUNT-I+1
      DO 32 J=1,NA
   32 ADD(ID)=ADD(ID)+P(I,J)
      RETURN
      END
```

FIGURE 5. THE POLYNOMIAL SIMPLIFICATION PROGRAM (Continued)

```
C          SAMPLE POLYNOMIAL PROBLEM DATA CARDS
 0         NA=0 → (A1, A2, ...)
 2  1      NP=2, KK=1
    5.E0   K(1)=5.0                                   (B)
 4         N1 = 4TH ORDER
    1.E0      -2.E0      2.E0      1.E0      -1.E0    (s^4 - 2s^3 + 2s^2 + s - 1)
 2         N2 = 2ND ORDER
    1.E0       0.E0      2.E0                         (s^2 + 2)
-------------------------------------------------------------------
 2         NA = 2 → (C1, C2, ...)
 2  0      NP, KK
 2         N1                                        (C1)
    1.E0       2.E0      -3.E0                       (s^2 + 2s - 3)
 1         N2
    1.E0       2.E0                                  (s + 2)
-------------------------------------------------------------------
 2  1      NP, KK
    2.E1   K(1) = 20.0
 3         N1                                        (C2)
    1.E0       0.E0      0.E0      -2.E0             (s^3 - 2)
 1         N2
    1.E0       1.E0                                  (s + 1)
-------------------------------------------------------------------
 2  0      NP, KK
 2         N2 = 2
    1.E0      -3.E0      1.E0                        (D)   (s^2 - 3s + 1)

        POLYNOMIAL 1 = (C1 + C2)
        POLYNOMIAL 2 = (s^2 - 3s + 1)
```

FIGURE 6. DATA FOR THE POLYNOMIAL SAMPLE PROGRAM

```
0          NA → (A1, A2, ...)
7  6       NP, KK
    1.E0    1.52E0     9.4E0      1.E1     5.56E5     4.E-1     CONSTANTS K(I)
2          N1
7.700E-5   8.520E-3      1.E0    Poly 1
1          N2
1.785E-2      1.E0             Poly 2
1          '
2.330E-2      1.E0             '
1          '
1.050E-3      1.E0             '          (B)        NUMERATOR
1          '
7.220E-4      1.E0             '
1          '
5.350E-5      1.E0             '
2          '
3.770E-5   8.580E-3      1.E0             '
-------------------------------------------------------------------
3          NA → (C1, C2, ...)
10 2       NP, KK
    8.8E0   1.080E4     CONSTANTS  K(I) = K(1), K(2)
1          N1
   4.E-4       1.E0             Poly 1
1          N2
    1.E0       0.E0             Poly 2
1          '
 4.93E-3       1.E0             '
1          '                                          DENOMINATOR
 4.97E-4       1.E0             '
1          '
 5.92E-3       1.E0             '
1          '                     '          (C1)
   2.5E0       1.E0             '
1          '
 1.05E-3       1.E0             '
1          '
 7.22E-4       1.E0             '
1          '
 5.35E-5       1.E0             '
2          '
 3.77E-5    8.58E-3      1.E0             '
-------------------------------------------------------------------
4  9       NP, KK
    1.E0    1.52E0     9.40E0     1.E1     7.98E0     1.2E0     9.E-1     1.6E0
  5.56E5                CONSTANTS  K(I) = K(1) ... K(9)
2          N1
 7.70E-5    8.52E-3      1.E0    Poly 1
1          N2
1.785E-2      1.E0             Poly 2     (C2)
1          '
 2.33E-2       1.E0             '
2          '
 4.03E-8    1.205E-6     1.E0             '
-------------------------------------------------------------------
```

FIGURE 7. DATA FOR SIMPLIFICATION OF THE TWELFTH ORDER POLYNOMIAL

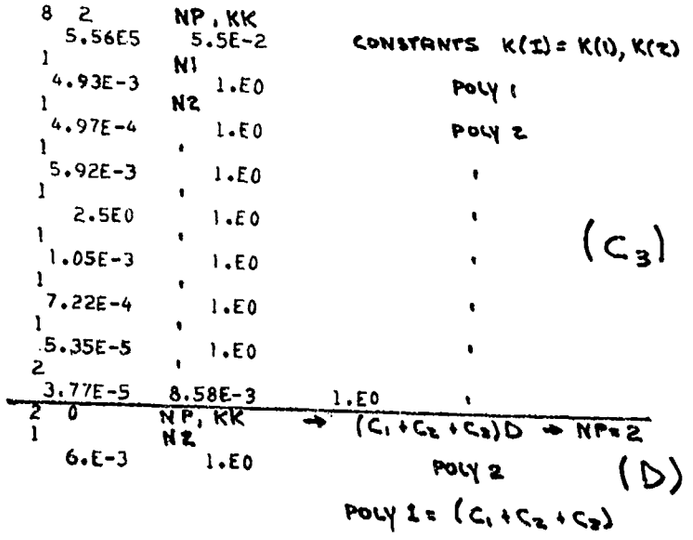


FIGURE 7. DATA FOR SIMPLIFICATION OF THE TWELFTH ORDER POLYNOMIAL (Continued)

```
C     CONTINUED FRACTION EXPANSION AND REDUCTION IN ORDER OF AN N-TH ORDER
C     TRANSFER FUNCTION
C
C        NN = TOTAL NUMBER OF TERMS IN THE NUMERATOR
C        A(1)...A(NN) = THE COEFFICIENTS OF THE NUMERATOR BEGINNING WITH
C              THE CONSTANT TERM AND INCREASING IN ORDER
C        ND = TOTAL NUMBER OF TERMS IN THE DENOMINATOR
C        B(1)...B(ND) = THE COEFFICIENTS OF THE DENOMINATOR BEGINNING WITH
C              THE CONSTANT TERM AND INCREASING IN ORDER
C
C        FINAL REDUCED ORDER TRANSFER FUNCTIONS ARE PRINTED AS
C              A0 + A1*S + A2*S**2 + A3*S**3 +... + AM*S**M
C              ---------------------------------------
C              B0 + B1*S + B2*S**2 + B3*S**3 +... + BN*S**N
C
      DIMENSION A(200),B(200),H(200)
      READ(5,500) NN
  500 FORMAT(I5)
      ND=NN+1
      READ(5,501) (A(I),I=1,NN)
      READ(5,501) (B(I),I=1,ND)
  501 FORMAT(4E20.6)
      L=0
    2 L=L+1
      H(L)=B(1)/A(1)
      WRITE(6,604) L,H(L)
  604 FORMAT(/2X,2HH(,I2,3H) =,1PE16.6)
      A(ND)=0.
      ND=ND-1
      DO 10 I=1,ND
      B(I)=A(I)
      I1=I+1
   10 A(I)=B(I1)-A(I1)*H(L)
    1 L=L+1
      H(L)=B(1)/A(1)
      WRITE(6,604) L,H(L)
      IF(ND.EQ.1) GO TO 3
      ND=ND-1
      DO 20 I=1,ND
      B(I)=A(I)
      I1=I+1
   20 A(I)=B(I1)-A(I1)*H(L)
      ND=ND+1
      B(ND)=A(ND)
      GO TO 2
    3 MM=2*NN
      CALL MRML(MM,H)
      STOP
      END
```

FIGURE 8. LISTING OF THE TRANSFER FUNCTION ORDER REDUCTION PROGRAM

```
      SUBROUTINE MRML(N,H)
      DIMENSION HR(30,30),TR(30,30),T(30,30),D(30),TL(30,30),C(30),H(30)
      DO 1 I = 1,N
      DO 1 J = 1,N
    1 HR(I,J) = 0.
      DO 10 J=1,N
      DO 20 L=1,J
      HR(L,L) = H(J+1-L)
      IF(L.EQ.N) GO TO 22
      IF(J+1-L.EQ.1) GO TO 21
   20 HR(L,L+1) = 1.
   21 J1=J+1
      DO 30 I=J1,N
   30 HR(I,I) = 1.
   22 IF(J.GT.1) GO TO 23
      DO 40 I1 = 1,N
      DO 40 J1 = 1,N
      T(I1,J1) = HR(I1,J1)
   40 HR(I1,J1) = 0.
      GO TO 10
   23 CALL MALTP (N,N,HR,N,T,TR)
      DO 50 I1 = 1,N
      DO 50 J1 = 1,N
      T(I1,J1) = TR(I1,J1)
   50 HR(I1,J1) = 0.
   10 CONTINUE
      DO 110 J=1,N
      DO 120 L=1,J
      IF(J.EQ.1)  GO TO 121
      HR(L,L) = H(J+1-L)
      IF(J+1-L.EQ.2)GO TO 121
  120 HR(L,L+1) = 1.
  121 DO 130 I=J,N
  130 HR(I,I) = 1.
      IF(J.GT.1) GO TO 123
      DO 140 I1 = 1,N
      DO 140 J1 = 1,N
      T(I1,J1) = HR(I1,J1)
  140 HR(I1,J1) = 0.
      GO TO 110
  123 CALL MALTP (N,N,HR,N,T,TL)
      DO 150 I1 = 1,N
      DO 150 J1 = 1,N
      T(I1,J1) = TL(I1,J1)
  150 HR(I1,J1) = 0.
  110 CONTINUE
      DO 200 I=1,N,2
      L = 0
      DO 210 J=I,N
      L = L+1
      D(L) = TR(I,J)
  210 C(L) = TL(I,J)
      NT =(N-I+1)/2+1
      NT1 = NT-1
      CALL ISE(NT1,D,C)
  200 CONTINUE
      RETURN
      END
```

FIGURE 8. LISTING OF THE TRANSFER FUNCTION ORDER REDUCTION PROGRAM (Continued)

```
      SUBROUTINE MALTP(N,M,A,L,B,C)
      DIMENSION A(30,30),B(30,30),C(30,30)
      DO 10 I=1,N
      DO 10 J=1,L
      S=0.0
      DO 10 K=1,M
      S=S+A(I,K)*B(K,J)
   10 C(I,J)=S
      RETURN
      END

      SUBROUTINE ISE(N,D,C)
      DIMENSION A(30),B(30),C(30),D(30),G(60),DD(30,30),
     1DN(30,30),T(3,30),F(30,30)
  600 FORMAT(//////)
  601 FORMAT(1H0,1P8E15.6)
  602 FORMAT(1H0,122H  ----------------------------------------------------
     1----------------------------------------------------------------
     1-----)
      N1=N+1
      WRITE(6,600)
      WRITE(6,601) (C(I),I=1,N)
   20 WRITE(6,602)
      WRITE(6,601) (D(I),I=1,N1)
      NX=N
      DO 50 I=1,2
      DO 51 J=1,N
      NJ=NX-2*(J-1)
      IF(NJ.LE.0) T(I,J) = 0.
   51 IF(NJ.GT.0) T(I,J)=D(NJ)
   50 NX=N+1
      DO 60 I=1,N
      DO 60 J=1,N
      DD(I,J)=0.
   60 DN(I,J)=0.
      L=0
      LL=0
      LJ=1
      DO 70 I=1,N
      L=L+1
      IF(L.EQ.3) LJ=LJ+1
      IF(L.EQ.3) L=1
      DO 71 J=LJ,N
      LL=LL+1
      DD(I,J)=T(L,LL)
   71 DN(I,J)=T(L,LL)
   70 LL=0
      N2=2*N
      DO 80 I=1,N2
   80 G(I)=0.
      L=0
      LL=0
      DO 81 I=1,N
      DO 82 J=1,N
      L=LL+J
   82 G(L)=C(J)*(-1.)**(J+1)*C(I)+G(L)
   81 LL=LL+1
```

FIGURE 8. LISTING OF THE TRANSFER FUNCTION ORDER REDUCTION PROGRAM (Continued)

```
      DO 90 J=1,N
      JJ=2*(N-J)+1
   90 DN(1,J)=G(JJ)
      CALL INVER(DN,N,F,0,DETN)
      CALL INVER(DD,N,F,0,DETD)
      XI=(-1.)**(N-1)/(2.*D(N1))*DETN/DETD
      RETURN
      END

      SUBROUTINE INVER (A,N,B,M,DET)
      DIMENSION A(30,30),B(30,30),IPVOT(30),INDEX(30,3),PIVOT(30)
      EQUIVALENCE (IROW,JROW),(ICOL,JCOL)
   57 DET=1.0
      DO 17 J=1,N
   17 IPVOT(J)=0
      DO 135 I=1,N
      T=0.0
      DO 9 J=1,N
      IF(IPVOT(J)-1) 13,9,13
   13 DO 23 K=1,N
      IF(IPVOT(K)-1) 43,23,81
   43 IF(ABS(T)-ABS(A(J,K))) 83,23,23
   83 IROW=J
      ICOL=K
      T=A(J,K)
   23 CONTINUE
    9 CONTINUE
      IPVOT(ICOL)=IPVOT(ICOL)+1
      IF(IROW-ICOL) 73,109,73
   73 DET=-DET
      DO 12 L=1,N
      T=A(IROW,L)
      A(IROW,L)=A(ICOL,L)
   12 A(ICOL,L)=T
      IF(M) 109,109,33
   33 DO 2 L=1,M
      T=B(IROW,L)
      B(IROW,L)=B(ICOL,L)
    2 B(ICOL,L)=T
  109 INDEX(I,1)=IROW
      INDEX(I,2)=ICOL
      PIVOT(I)=A(ICOL,ICOL)
      DET=DET*PIVOT(I)
      A(ICOL,ICOL)=1.
      DO 205 L=1,N
  205 A(ICOL,L)=A(ICOL,L)/PIVOT(I)
      IF(M) 347,347,66
   66 DO 52 L=1,M
   52 B(ICOL,L)=B(ICOL,L)/PIVOT(I)
  347 DO 134 LI=1,N
      IF(LI-ICOL) 21,134,21
   21 T=A(LI,ICOL)
      A(LI,ICOL)=0.
      DO 89 L=1,N
   89 A(LI,L)=A(LI,L)-A(ICOL,L)*T
      IF(M) 134,134,18
```

FIGURE 8. LISTING OF THE TRANSFER FUNCTION ORDER REDUCTION PROGRAM (Continued)

```
 18 DO 68 L=1,M
 68 B(LI,L)=B(LI,L)-B(ICOL,L)*T
134 CONTINUE
135 CONTINUE
222 DO 3 I=1,N
    L=N-I+1
    IF(INDEX(L,1)-INDEX(L,2)) 19,3,19
 19 JROW = INDEX(L,1)
    JCOL = INDEX(L,2)
    DO 549 K = 1,N
    T = A(K,JROW)
    A(K,JROW) = A(K,JCOL)
    A(K,JCOL) = T
549 CONTINUE
  3 CONTINUE
 81 RETURN
    END
```

FIGURE 8. LISTING OF THE TRANSFER FUNCTION ORDER REDUCTION PROGRAM (Continued)

12

3.177651E+07	1.908990E+06	4.494978E+04	5.801820E+02.
4.811424E+00	2.424566E-02	7.229334E-05	8.449636E-08
3.340163E-11	1.556011E-15	0.0E-00	0.0E-00
1.095481E+09	6.115740E+07	1.493007E+06	1.964881E+04
1.624716E+02	7.929650E-01	2.440315E-03	5.966089E-06
7.968180E-09	5.703077E-12	2.083613E-15	3.327238E-19
1.264749E-23			

FIGURE 9. DATA FOR THE ORDER REDUCTION PROGRAM

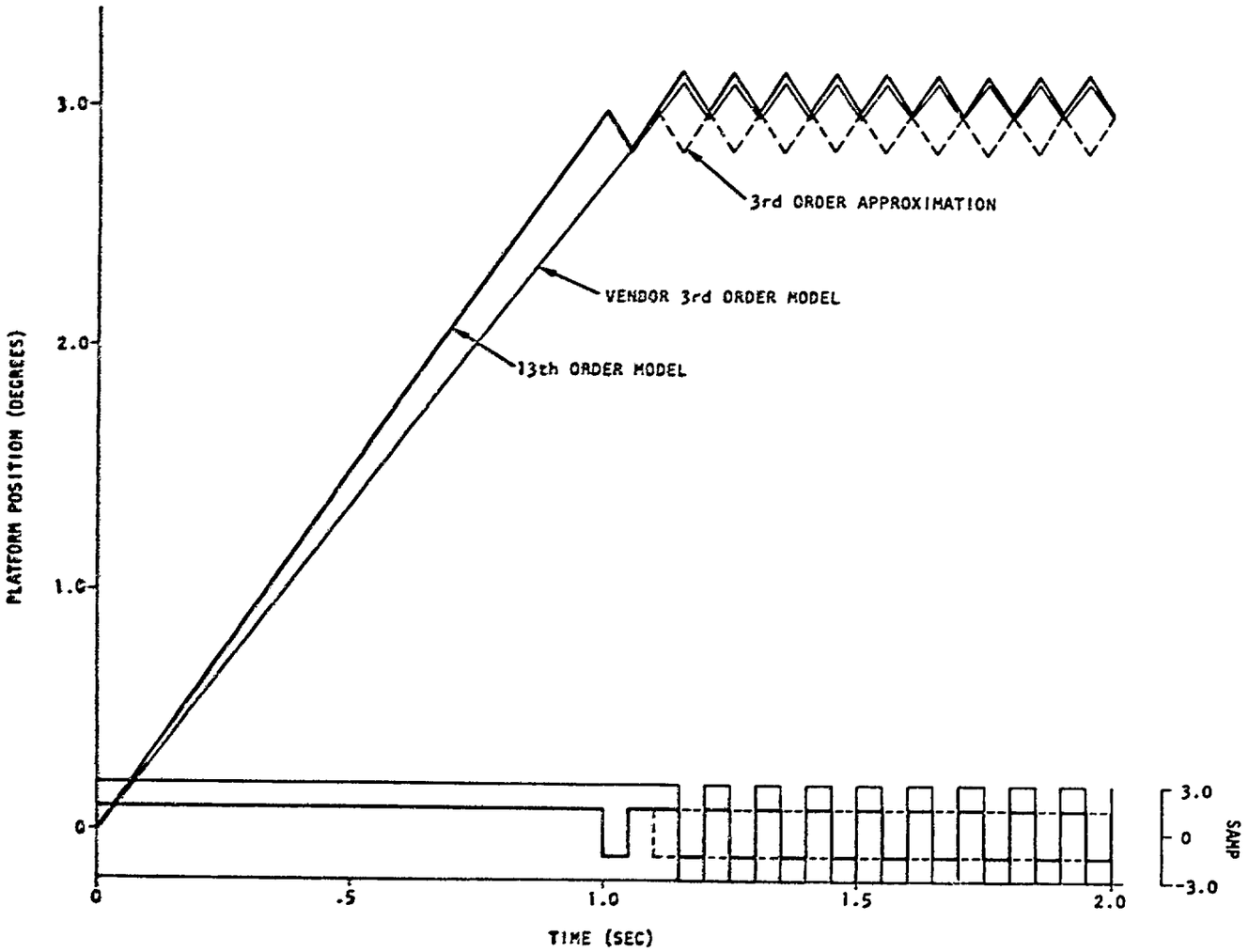


FIGURE 10. TRACKING RESPONSES OF THE 13th AND 3rd ORDER IN-HOUSE SEEKER MODELS, THE VENDOR MODEL, AND THE TRACK COMMAND (SAMP) FUNCTIONS

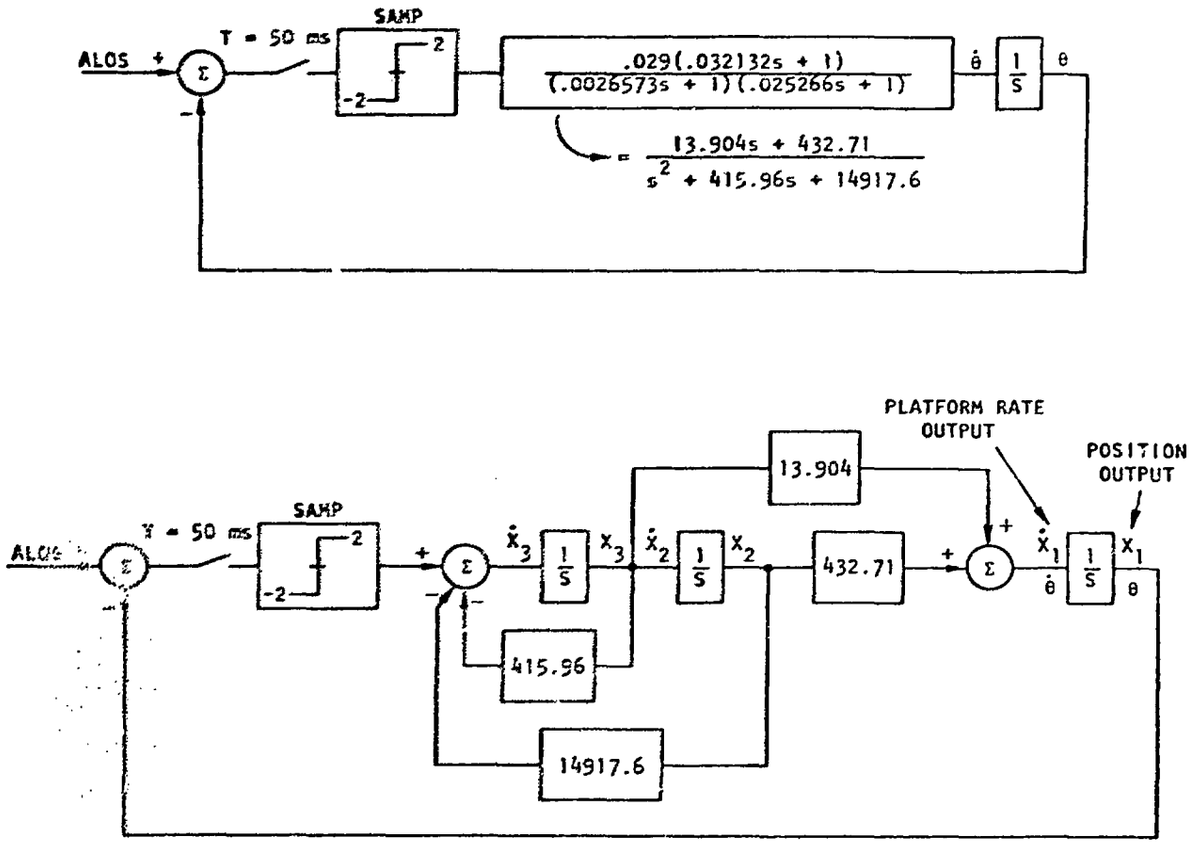


FIGURE 11. BLOCK DIAGRAM AND STATE VARIABLE MODEL OF THE IN-HOUSE THIRD ORDER RATE LOOP APPROXIMATION

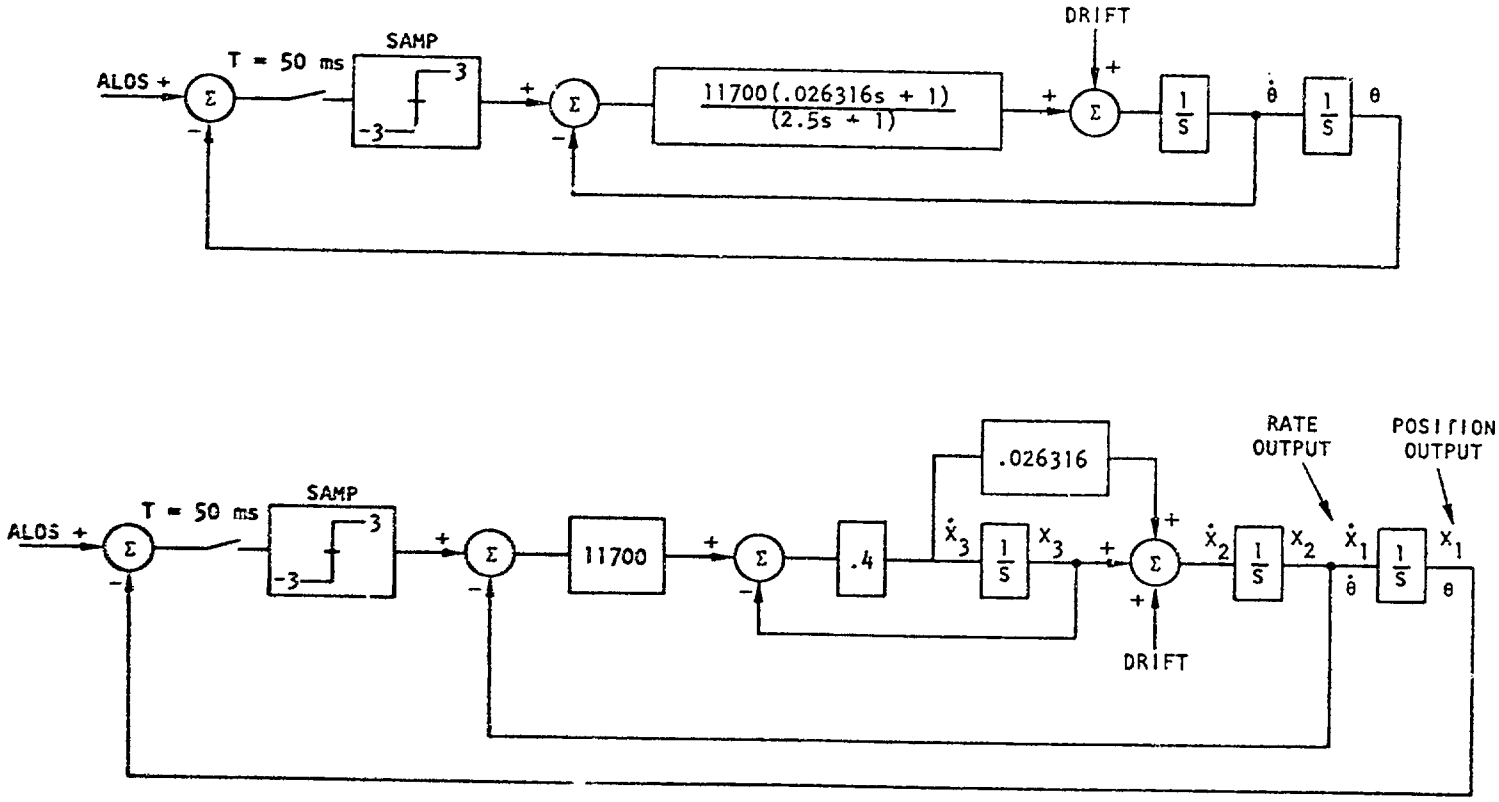


FIGURE 12. BLOCK DIAGRAM AND STATE EQUATION MODEL OF THE VENDOR THIRD ORDER SYSTEM APPROXIMATION